MUSIKA: A multichannel multi-sink data gathering algorithm in wireless sensor networks

Ridha Soua, Erwan Livolant, Pascale Minet

MUSIKA: a Multichannel Multi-Sink Data Gathering Algorithm in Wireless Sensor Networks

Ridha Soua, Erwan Livolant, Pascale Minet
Inria Rocquencourt, 78153 Le Chesnay cedex, France
Email: firstname.name@inria.fr

***Abstract*—A typical task in wireless sensor networks (WSNs) is to collect data from sensor nodes towards one or many sinks in a multi-hop convergecast structure. In this paper, we focus on the data gathering problem with differentiated traffic, each addressed to a specific sink in multichannel WSNs. In order to find a collision-free optimized multichannel time slot assignment that minimizes the data gathering cycle, we propose a centralized traffic-aware algorithm called MUSIKA. We formulate the problem as a linear program and compute the optimal time needed for a raw data convergecast in an illustrative example. More generally, we run simulations on various network topologies to evaluate the performance of MUSIKA in terms of cycle length, maximum buffer size and slot reuse ratio for different use cases: redundant functional processing chains, different application functionalities per sink.**

***Index Terms*—multichannel wireless sensor networks, multi-sink, convergecast, time slot assignment, optimized schedule**

I. Context

Tiny and inexpensive sensor nodes with wireless communication capability, small battery and limited processing power are deployed for monitoring homes, nuclear power plants, aircrafts, etc. This control process imposes the need to gather data from sensors and deliver them to central entities, usually called sinks. Such a communication scheme is also called convergecast.

Under heavy traffic conditions, contention-based protocols suffer from collisions and non deterministic delays. In contrast, the contention-free deterministic scheduling is energy efficient by avoiding collisions and allowing low power sensors to turn off their radio in time slots not assigned to them. Moreover, minimizing the number of slots in the contention-free cycle improves the network performances by minimizing the maximum packet delay, increasing throughput and finally reducing the activity period of nodes.

In single channel WSNs, the data rate of contention-free scheduling is significantly limited by co-channel interferences, particularly in dense deployments. Furthermore, multichannel paradigm [1] [2] allows parallel transmissions and thus ensures a higher throughput. Therefore, we tackle in this paper the problem of time slot assignment for convergecast in multichannel WSNs.

Besides, the use of multiple sinks ensures:

1) A more reliable data gathering. This property is expected especially for critical information (path diversity), since in many deployments the channel used by the WSN may encounter perturbations or noise.
2) Energy efficiency by decreasing both the load and the energy consumption of nodes close to the sink. Indeed, these nodes must forward a higher traffic toward the sink, which is a severe threat to network lifetime. The existence of several sinks enables a better load balancing between nodes.
3) Sinks running different functionalities of the application considered. This allows a higher flexibility in the mapping of application functionnalities on wireless nodes. Indeed, this mapping may depend on several factors such as node location, desired redundancy degree and may take into account heterogeneous application requirements with regard to expected functionalities.

In this paper, unlike studies that focus only on the time slot assignment problem, i.e. how to send data collected by multiple sources to a common sink and if possible in a minimum of time, we focus on a multi-sink multichannel context with a dedicated traffic per sink.

Mainly in this paper, we formalize the multichannel multi-sink convergecast problem in WSNs. We then propose our algorithm MUSIKA. Finally, performances in terms of cycle length, delivery delay and buffer size of the MUSIKA algorithm are evaluated by simulation in Section VI.

II. Motivation

One limitation of many proposed time slot assignments for convergecast is that they do not involve multichannel paradigm. The throughput requirements of many applications of WSNs is difficult to meet with a single wireless channel.

In multichannel WSNs, we distinguish two issues: channel assignment and node scheduling for transmissions and receptions. About the former issue, readers can refer to [1] for further details. Concerning the latter issue, since contention-free scheduling techniques prevent major sources of inefficiency (idle listening, overhearing, and collisions), we limit the scope of studied works on multichannel contention-free based scheduling protocols.

Tree-based Multi-Channel Protocol (TMCP) [2] supports data collection traffic by partitioning the network into multiple subtrees and then assigns different channels to different subtrees. Hence, inter-tree interference is minimized without the need for time synchronization.

978-1-4673-2480-9/13/$31.00

Zhang et al. study the joint link scheduling and channel assignment for convergecast in WirelessHart based networks [3]. They present an optimal joint time and channels scheduling algorithm with time complexity $O(N^2)$, where N is the number of sensor nodes.

In [4], PIP, a joint TDMA-FDMA based bulk transfer protocol was proposed. Nevertheless, PIP presents scalability limitation because the sink can only be involved in two connections simultaneously.

Incel et all [5], consider the case where all interfering links are removed with necessary number of channels. They propose JFTSS a Joint Frequency Time Slot Scheduling algorithm that provides the smallest number of slots for any network topology where the routing tree has an equal number of nodes on each branch.

In [6], authors focus on time slot assignment in a multi-sink single hop UWB WSNs which is formulated as a linear programming problem. They also implement a heuristic to improve both throughput and fairness. They show that it is deemed scalable with multiple sinks. A drawback of this work is that it is limited to single hop networks.

None of the above related works about time slot assignment for convergecast deals with multi-hop multichannel multi-sink WSNs. That is why, this paper proposes on the one hand a linear programming formalization of the problem and on the other hand a deterministic contention-free based algorithm called MUSIKA for convergecast in multichannel multi-sink WSNs.

III. Problem formalization

In this section, we present a formalization of the multi-sink slot assignment problem according to the assumptions listed below. We are looking for a multichannel slot assignment of minimum length ensuring that there are no two conflicting nodes transmitting simultaneously on the same channel.

A. Assumptions

◦ **A1. Available channels:** For the sake of simplicity, we assume that at each node, $MaxChannel > 1$ channels are available. We assume that network connectivity is ensured on any of these channels and any node has the same neighbors on all channels.

◦ **A2. Single radio interface:** All nodes have a single radio interface that can be tuned on the selected channel.

◦ **A3. Differentiated traffic:** Each traffic generated by a source node is tagged with its destination sink and must be transmitted to this sink. With each traffic is associated its importance degree from the application point of view. A traffic class groups all traffic with the same importance degree and the same sink as final destination. Each node maintains a FIFO queue per traffic class. In this paper, we consider the general case where two sinks may have traffic with the same importance degree or different importance degrees.

◦ **A4. Raw data convergecast:** Each node transmits its own data to its parent in the data gathering tree rooted at the sink corresponding to this traffic. It forwards the data received from its children, without aggregation.

For the sake of simplicity, we adopt also the following assumptions:

◦ **A5. Slot size:** We assume that the slot size enables the transmission of a single packet corresponding to the data generated by a node. Moreover, each unicast transmission is acknowledged in the time slot of the sender (i.e. immediate acknowledgment).

◦ **A6. Conflicting nodes:** Two nodes are said *conflicting* if and only if they cannot transmit in the same time slot on the same channel. By definition, $Conflict(u)$ is the set of nodes conflicting with u. This set is an input of MUSIKA.

◦ **A8. Topology links:** We also assume that the only topology links are those represented in the convergecast trees.

◦ **A9. Ideal environment:** In this paper, we assume there is neither message loss, nor node failure.

The assumption $A9$ can be relaxed by considering packet retransmission to recover from packet losses. In this case, the amount of traffic should take into account these packets retransmissions. These latter can be evaluated considering an estimated packet loss rate.

Similarly assumption $A5$ can be relaxed considering that the slot size allows the transmission of $p > 1$ packets. In such a case, the traffic demand should be mapped into a slot demand.

B. Model

The network is modeled as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of vertices representing the nodes of the network and $\mathcal{E}$ is the set of edges representing the communication links between nodes.

Let $\mathcal{V}_s$ be the set of sinks, with $\mathcal{V}_s \subset \mathcal{V}$. For each node $v \in \mathcal{V}$ and $s \in \mathcal{V}_s$ with $v \neq s$, we define $p_{v,s}$ the number of packets that v generates at each cycle and has to transmit towards the sink s. Moreover, for any node $v \in \mathcal{V}$, let $Conflict(v)$ be the set of conflicting nodes that interfere with v when transmitting on the same channel. Let $\mathcal{E}^+(v)$ denote the set of links through which a node $v \in \mathcal{V}$ can transmit. Let $\mathcal{E}^-(v)$ be the set of links through which a node $v \in \mathcal{V}$ can receive.

Let $\mathcal{C}$ be the set of channels usable for any transmission. The contention-free cycle is composed of at most T_{max} slots, where T_{max} denotes the maximum length of the cycle.

We define $a_{e,v,s,c,t}$ the activity of a link $e \in \mathcal{E}$ transferring a packet originated from $v \in \mathcal{V}$ towards $s \in \mathcal{V}_s$ on the channel $c \in \mathcal{C}$ in the slot t, ie $a_{e,v,s,c,t} = 1$ if and only if there is a transmission of a packet originated from v to s on the link e on the channel c in the time slot t and $a_{e,v,s,c,t} = 0$ otherwise.

Furthermore, let u_t be the use of a slot t, in other words $u_t = 1$ means that there is at least one link activity on at least one channel in the slot t and $u_t = 0$ denotes an empty slot. The objective is to minimize the number of slots t used in the cycle:

$$min \sum_{t=1}^{T_{max}} u_t$$

This objective is subject to:

$$a_{e,v,s,c,t} \leq u_t \quad \forall e \in \mathcal{E}, \forall v \in \mathcal{V}, \forall s \in \mathcal{V}_s, \forall c \in \mathcal{C}, t \leq T_{max}, \tag{1}$$

$$\sum_{o \in \mathcal{V}} \sum_{s \in \mathcal{V}_s} a_{e,o,s,c,t} + \sum_{o \in \mathcal{V}} \sum_{s \in \mathcal{V}_s} a_{e',o,s,c,t} \leq 1 \quad \forall v \in \mathcal{V}, \forall e \in \mathcal{E}^+(v), \forall w \in Conflict(v), \forall e' \in \mathcal{E}^+(w), \forall c \in \mathcal{C}, t \leq T_{max} \tag{2}$$

$$\sum_{e \in \mathcal{E}^+(v)} \sum_{c \in \mathcal{C}} \sum_{t=1}^{T_{max}} a_{e,v,s,c,t} = p_{v,s} \quad \forall s \in \mathcal{V}_s, \forall v \in \mathcal{V} \setminus \{s\} \tag{3}$$

$$\sum_{e \in \mathcal{E}^+(v)} \sum_{c \in \mathcal{C}} \sum_{t=1}^{T_{max}} a_{e,w,s,c,t} = \sum_{e \in \mathcal{E}^-(v)} \sum_{c \in \mathcal{C}} \sum_{t=1}^{T_{max}} a_{e,w,s,c,t} \quad \forall s \in \mathcal{V}_s, \forall v \in \mathcal{V} \setminus \{s\}, \forall w \in \mathcal{V} \setminus \{v\} \tag{4}$$

$$\sum_{e \in \mathcal{E}^-(s)} \sum_{v \in \mathcal{V}} \sum_{c \in \mathcal{C}} \sum_{t=1}^{T_{max}} a_{e,v,s,c,t} = \sum_{w \in \mathcal{V} \setminus \{s\}} p_{w,s} \quad \forall s \in \mathcal{V}_s \tag{5}$$

$$\sum_{e \in \mathcal{E}^+(v)} \sum_{c \in \mathcal{C}} a_{e,v,s,c,t} \leq p_{v,s} - \sum_{e \in \mathcal{E}^+(v)} \sum_{c \in \mathcal{C}} \sum_{t'=1}^{t} a_{e,v,s,c,t'} \quad \forall s \in \mathcal{V}_s, \forall v \in \mathcal{V} \setminus \{s\}, t \leq T_{max} \tag{6}$$

$$\sum_{e \in \mathcal{E}^+(v)} \sum_{c \in \mathcal{C}} a_{e,v,s,c,t} \leq \sum_{e \in \mathcal{E}^+(v)} \sum_{c \in \mathcal{C}} \sum_{t'=1}^{t} a_{e,w,s,c,t'} - \sum_{e \in \mathcal{E}^-(v)} \sum_{c \in \mathcal{C}} \sum_{t'=1}^{t} a_{e,w,s,c,t'} \quad \forall s \in \mathcal{V}_s, \forall v \in \mathcal{V} \setminus \{s\}, \forall w \in \mathcal{V} \setminus \{v\}, t \leq T_{max} \tag{7}$$

Constraint 1 binds the use of a time slot to at least the activity of one link on any channel in this slot. Constraint 2 guarantees that two conflicting nodes v and w do not transmit on the same channel in the same time slot.

Constraint 3 ensures that for each sink s, any non-sink node v transmits during the cycle all the packets it has generated for s. Constraint 4 expresses that for each sink s, any intermediate node v forwards towards s all the received packets destinated to s. Constraint 5 ensures that each sink s receives all the packets generated in the WSN with final destination s.

Constraint 6 expresses the rule that a node v transmits a packet towards a sink s at the slot t if and only if its buffer of this traffic is not empty. Constraint 7 guarantees that any sink s forwards the traffic addressed to another sink it receives.

IV. MUSIKA: MULTI-SINK SLOT ASSIGNMENT

In this section, we present MUSIKA, a centralized raw data convergecast scheduling algorithm for multichannel multi-sink WSNs. MUSIKA is based on our previous work [7]. However, this latter did not take into account the existence of multiple sinks and traffic differenciation. Therefore, we propose a novel solution to address this new context.

A. Principles

MUSIKA proceeds slot by slot to build the multichannel multi-sink schedule, applying the following rules:

R1. Only nodes having at least one packet to transmit compete for the current time slot. They are ordered according to their decreasing priority. Let $\mathcal{N}$ be this ordered set.

R2. The competing node in $\mathcal{N}$ with the highest priority is selected first.

R3. A node is allowed to transmit in the current slot if and only if:
   1) this node and its parent in the data gathering tree corresponding to the packet to transmit have an available radio interface;
   2) there exists a channel where this node does not conflict with nodes already scheduled in this slot.

R4. A node allowed to transmit in the current slot will transmit the first packet in the FIFO queue of the traffic class with the highest importance degree. If several traffic classes have the same importance degree, the first packet of the longest queue in these traffic classes will be chosen.

R5. The next node to be selected is the next one in $\mathcal{N}$. It is allowed to transmit according to rule R3 and will transmit its packet selected according to rule R4. And so on until all nodes in $\mathcal{N}$ have been checked for a possible transmission.

In the illustrative example given in Section V as well as in the performance evaluation reported in Section VI, we assign priorities to nodes as follows. The priority of a node is computed taking into account:

1) the number of packets present in its queues, to avoid buffer saturation;
2) the sum for each data gathering tree of the number of packets its parent should receive in a cycle, to favor nodes with a high load;
3) and the classes of packets to be transmitted by the node in the current slot, to provide traffic differentiation if required by the application.

Furthermore, in order to obtain a strong traffic differentiation, we require that for any sink s the priority of any packet in a class i is higher than the priority of any packet in a class j with a strictly less importance degree ($j < i$). For simplicity sake, we assume that classes are ordered according to a non-decreasing importance degree. That is why, we define $prioClass_i$ for any class i and $prio_u$ for any node u as follows:

$$prioClass_i = \prod_{j<i} (1 + sinkRcv_j^2)$$

where $sinkRcv_j$ is the total number of packets that should be received by s for flows belonging to class j, where s is the sink associated with class j. By convention, $prioClass_1$=1.

$$prio_u = \sum_i [prioClass_i * \sum_{f \in i} (remPckt_f * parentRcv_f)]$$

where $prioClass_i$ is the priority of class i to which a flow f present on node u belongs to, $remPckt_f$ means the number

of packets of flow f the node u has in its buffer at the current iteration and $parentRcv_f$ is the total number of packets of flow f that the parent of node u has to receive in a cycle. Notice that in $prio_u$ two factors $prioClass_i$ and $parentRcv_f$ are static during the cycle whereas the factor $remPckt_f$ depends on the size of the buffer queue and hence depends on the slot considered.

B. Algorithm

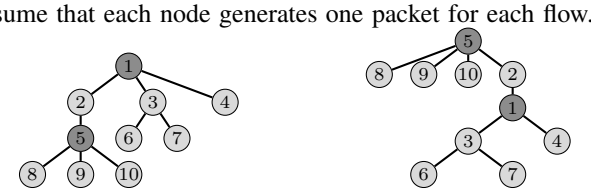
The channel selection strategy given in the algorithm below is greedy but other strategies like Round Robin are also possible to achieve a better load balance.

Algorithm 1 MUSIKA algorithm

Input: cl_{max} traffic classes with their associated gathering trees, each node u has one available radio interface i_u, $n_{channel}$ channels, d_u^f packets of flow f to transmit and a set of conflicting nodes $Conflict(u)$.
Output: The multi-sink scheduling of nodes in the contention-free cycle
/* Initialization phase */
$\forall cl$, $prioClass_{cl} \leftarrow \prod_{cl'<cl}(1 + sinkRcv_{cl'}{}^2)$
$\forall u$, $prio_u \leftarrow \sum_i [prioClass_{cl} * \sum_{f \in cl}(remPckt_f * parentRcv_f)]$
$t \leftarrow 0$ // current time slot
/* Scheduling phase */
while $\sum_f \sum_u d_u^f$ **do** // there are packets to transmit
  $\forall u$, $i_u \leftarrow True$ // u has an available radio interface
  $\forall c = 1..n_{channel}$, $conflict_c \leftarrow \emptyset$ // initialize conflicting nodes on channel c
  $\forall cl$, $N_{cl} \leftarrow$ list of nodes having data to transmit and sorted according to their priorities in the class cl.
  $t \leftarrow t + 1$
  /* Assignment of slot t */
  while $\cup_{cl} N_{cl} \neq \emptyset$ **do**
    $Tx \leftarrow False, nChannelReached \leftarrow False$
    repeat
      Select the node v with the highest priority in $\cup_{cl} N_{cl}$
      $\forall cl$, $N_{cl} \leftarrow N_{cl} \setminus \{v\}$
      Select the flow f with the highest priority on node v
    until i_v and $i_{parent(v)}$ // this node and its parent for flow f have an available interface
    $c \leftarrow 1$ // selected channel
    repeat
      if $v \notin conflict_c$ **then**
        Node v transmits in slot t on the channel c
        $d_v^f \leftarrow d_v^f - 1$
        $d_{parent(v)}^f \leftarrow d_{parent(v)}^f + 1$
        $i_v \leftarrow False$
        $i_{parent(v)} \leftarrow False$
        $conflict_c \leftarrow conflict_c \cup Conflict(v)$
        Update $prio_v$ and $prio_{parent(v)}$
        $Tx \leftarrow True$
      else
        if $c < n_{channel}$ **then**
          $c \leftarrow c + 1$ // change of selected channel
        else
          $nChannelReached \leftarrow True$
        end if
      end if
    until Tx || $nChannelReached$
  end while
end while

V. Illustrative example

The problem of optimal multichannel slot assignment is solved with the GPLK (GNU Linear Programming Kit) [8] solver based on the model presented in the section III. We consider an illustrative multichannel network topology with two sinks (see Figures 1(a) and 1(b)). There are exactly two traffic types and one flow per traffic type, denoted f_1 and f_2. The optimal time needed for a raw data convergecast is computed considering a single traffic (f_1 or f_2) and then both types of traffic f_1 and f_2 differentiated by their destination sink. These traffic types can have different importance degrees from the application point of view. For simplicity sake, we also assume that each node generates one packet for each flow.



(a) (b)

Fig. 1. The two tree topologies of the network.

We use GLPK to compute first the minimum number of slots required by each flow taken separately. We obtain 9 slots for f_1 and 11 slots for f_2. For the minimum number of slots required by the two flows, GLPK gives 20 slots and the optimal schedule provided by GLPK is illustrated by Figure 2. In the figures representing schedules, we adopt the following convention: the transmission of a packet of flow f_1 is represented on a white background, whereas this of flow f_2 appears on a black background. The main number within the cell indicates the origin of the packet, whereas the index number indicates the channel used. Notice that only two channels are needed. In Figure 2, we observe that despite the parallelism of transmissions between the different flows, the optimal cycle length for the two flows is equal to the sum of the optimal cycle length for each flow taken separately.

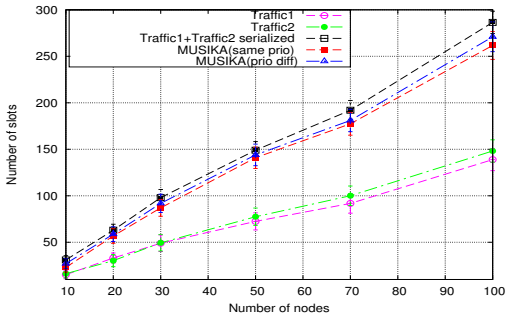
When the flows have the same priority, MUSIKA provides the following schedule illustrated by Figure 3. We notice that the number of slots is optimal, equal to 20 slots. We observe also that the transmissions of flows f_1 and f_2 are interleaved.

When the flows have different priorities, MUSIKA gives the schedule illustrated by Figure 4. We notice that flow f_1 that belongs to the highest priority class completes in the slot 9, exactly as if it were alone in the WSN, unlike previously where it completed in slot 20.

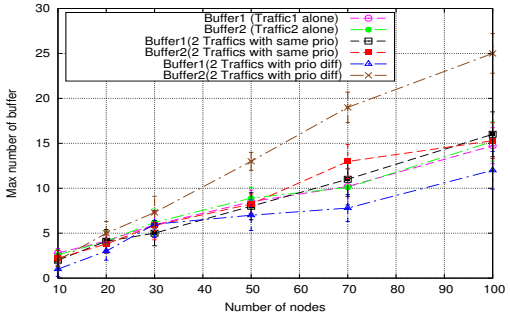
VI. Performance evaluation

We use a simulation tool based on GNU Octave [9] to evaluate the performances of MUSIKA in various topologies of WSNs. The number of nodes vary from 10 to 100. All nodes have a single radio interface. We assume that the only links are those belonging to the tree. In these simulations, $Conflict(u)$ is the set of one-hop and two-hop neighbors of u, for any node u. For each data gathering tree considered, the number of children per node is less than or equal to 3. Each result depicted on a figure is the average of 20 simulation runs.

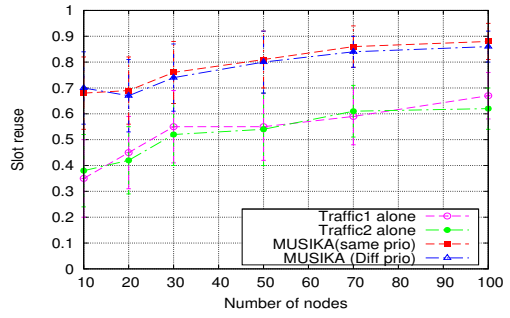
In the first series of experiments, we evaluate the average number of slots needed to complete convergecast. We distinguish two cases: both traffics have the same priority and a traffic has a higher priority than the other. Results are depicted in Figure 5(a). It appears that the number of slots required by MUSIKA in both cases (same and different priorities) is less than the number of slots where traffic 1 and traffic 2 are serialized: traffic 1 served before traffic 2. MUSIKA reduces the number of slots by first optimizing the scheduling of the first traffic and then applying spatial reuse to schedule the second traffic. For example in the 100 nodes configuration, as depicted in Figure 5(a), MUSIKA needs only 261 slots while a serial schedule of the two flows needs 288 slots, providing a gain of about 10%.



(a) Number of required slots



(b) Maximum buffer size



(c) Slot reuse ratio

Fig. 5. MUSIKA performances.

In the second series of experiments, we also evaluate the maximum number of buffers required in a node during a contention-free cycle. Figure 5(b) shows that when both traffics have the same priority, MUSIKA requires a number of buffers close to the number of buffers required when only one traffic is present. This can be explained by the basic priority of nodes used in MUSIKA which favors nodes having longer buffer queues to transmit.

In the third series of experiments, we consider the slot reuse ratio defined by the fraction of slots where at least there is two transmissions divided by the total number of slots. As illustrated by Figure 5(c), MUSIKA achieves the higher slot reuse ratio in the two cases of traffic priority. As shown in Figure5(c), for 100 nodes the slot reuse ratio is equal to 0.88 (two traffics with the same priority) while this ratio is equal to 0.68 when there is only one traffic. That can be explained by the fact that MUSIKA takes benefit of the spatial reuse concept when computing the schedules. Undoubtedly, the slot reuse ratio significantly reduces the end-to-end latency without a penalty in energy efficiency.

VII. Conclusion

In this paper, we aimed at deriving collision-free schedules for raw data convergecast with minimum latency in multi-sink multichannel WSNs. We identify two main reasons for the existence of several sinks: on the one hand redundancy of sinks improves robustness of data gathering and on the other hand, different application functionalities may be distributed on the sinks, explaining why they may have different importance degrees. To achieve this objective, we formulate the problem as a linear programming problem, aiming at deriving the smallest cycle length. Then, we propose the MUSIKA algorithm to obtain a collision-free schedule with the smallest frame length. This algorithm provides traffic differentiation if required by the application to reflect different importance degrees of traffic. From the simulation results, we conclude that MUSIKA shows its merit by taking advantage of spatial reuse to assign any slot to non-conflicting transmitters in both traffics, thus reducing the cycle length. Furthermore, the maximum number of buffers needed on a node is optimized with MUSIKA.

References

[1] R. Soua, P. Minet, *A survey on multichannel assignment protocols in wireless sensor networks*, In Proc. IFIP Wireless Days, Niagara Falls, Ontario, Canada, October, 2011.

[2] Y. Wu, J. Stankovic, T. He, S. Lin, *Realistic and efficient multi-channel communications in wireless sensor networks*, In Proc. INFOCOM'08, Phoenix, Arizona, April 2008.

[3] H. Zhang, P. Soldati, M. Johansson, *Optimal Link scheduling znd channel Assignment for convergecast in linear WirelessHART Networks*, In Proc. WiOPT'09, Seoul, Korea, June 2009.

[4] B. Raman, K. Chebrolu, S. Bijwe, V. Gabale, *PIP: A Connection-Oriented, Multi-Hop, Multi-Channel TDMA-based MAC for High Throughput Bulk Transfer*, In Proc. Sensys'10, Zurich, Switzerland, November 2010.

[5] O. D. Incel, A. Gosh, B. Krishnamachari, K. Chintalapudi, *Fast data Collection in Tree-Based Wireless Sensor Networks*, IEEE Transactions on Mobile computing, vol. 1, pp. 86-99, 2012.

[6] H. Tan, M-C. Chan, P-Y. Kong, C-K. Tham, *A Resource Allocation Scheme for TH-UWB Networks with Multiple Sinks*, In Proc. WCNC'08, Las Vegas, Nevada, March 2008.

[7] R. Soua, P. Minet, E. Livolant, *MODESA: an optimal multichannel slot assignment for raw data convergecast in wireless sensor networks*, In Proc. IPCCC 2012, Austin, Texas, December 2012.

[8] http://www.gnu.org/software/glpk/

[9] http://www.gnu.org/software/octave/

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$1 \rightarrow 2$		3_2				4_1						1_1			6_2			7_1		
$2 \rightarrow 1$					2_1		5_2		8_1				10_2							9_1
$2 \rightarrow 5$			3_2	2_2							4_2			1_2			6_2		7_2	
$3 \rightarrow 1$	3_2						6_2			3_1			6_2		7_1	7_1				
$4 \rightarrow 1$			4_1																4_2	
$5 \rightarrow 2$	5_1							8_2		10_2						9_1				
$6 \rightarrow 3$			6_2	6_2																
$7 \rightarrow 3$										7_2		7_2								
$8 \rightarrow 5$							8_1												8_2	
$9 \rightarrow 5$															9_2			9_1		
$10 \rightarrow 5$		10_2			10_2															

Fig. 2. The optimal multi-sink slot assignment obtained with the GLPK solver for the illustrative example.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$1 \rightarrow 2$		1_1		3_1									6_1		7_1		4_1			
$2 \rightarrow 1$						2_1		5_1		8_1		9_1								10_1
$2 \rightarrow 5$	2_1		1_1		3_2									6_1		7_1		4_1		
$3 \rightarrow 1$	3_2		6_2		3_1		6_1		7_1					7_2						
$4 \rightarrow 1$											4_1					4_2				
$5 \rightarrow 2$							5_1		8_1		9_1								10_1	
$6 \rightarrow 3$		6_1		6_1																
$7 \rightarrow 3$						7_1		7_1												
$8 \rightarrow 5$		8_1						8_2												
$9 \rightarrow 5$				9_1						9_2										
$10 \rightarrow 5$						10_2						10_2								

Fig. 3. The multi-sink slot assignment obtained with MUSIKA for the illustrative example with flows f_1 and f_2 having the same priority.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
$1 \rightarrow 2$											1_1		3_1		6_1		7_1		4_1	
$2 \rightarrow 1$	2_1		5_1		8_1		9_1		10_1											
$2 \rightarrow 5$										2_2		1_1		3_1		6_1		7_1		4_1
$3 \rightarrow 1$		3_1		6_1		7_1				3_1		6_2		7_2						
$4 \rightarrow 1$								4_1								4_2				
$5 \rightarrow 2$		5_1		8_1		9_1		10_1												
$6 \rightarrow 3$	6_1				6_1															
$7 \rightarrow 3$			7_1				7_1													
$8 \rightarrow 5$	8_2						8_2													
$9 \rightarrow 5$			9_2						9_2											
$10 \rightarrow 5$					10_2						10_1									

Fig. 4. The multi-sink slot assignment obtained with MUSIKA with flow f_2 represented in black belongs to a class of less importance degree than flow f_1 represented in white.